



Unidad 02: Fundamentos Rational Functional Tester

Equipo de Profesores del Curso



Logro de la Unidad de Aprendizaje

- Al término de la unidad, el alumno crea y agrega características más avanzadas a los scripts de pruebas funcionales a partir de los casos de prueba de su proyecto final.



TEMA 03: Introducción al Rational Functional Tester

**Curso: Pruebas de software
Equipo de Profesores del Curso**



Temario

1. Arquitectura de Rational Functional Tester
2. Configuración del entorno de pruebas
3. Configuración de aplicaciones Java a probar



1. Arquitectura de RFT

Almacenamiento de activos de pruebas

- Java
- XML

Test Logs

Almacenamiento de resultados de pruebas

- HTML, Texto, XML
- RQM

Grabación de pruebas funcionales

- Scripts
- Objetos de pruebas
- Datos de prueba
- PV

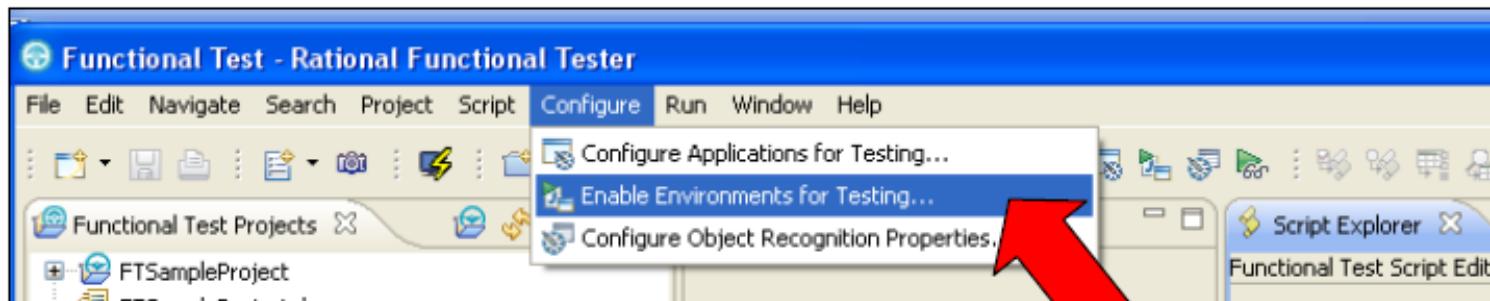
Ejecución de pruebas

- Pruebas funcionales
- Pruebas de Regresión

Integración con otras aplicaciones

- RQM, RCQ, RCC, JUnit, RPT, RAD

2. Configuración del entorno de pruebas



Configurar los navegadores web

(página 57 del manual)

Configurar entornos de Java

(página 59 del manual)

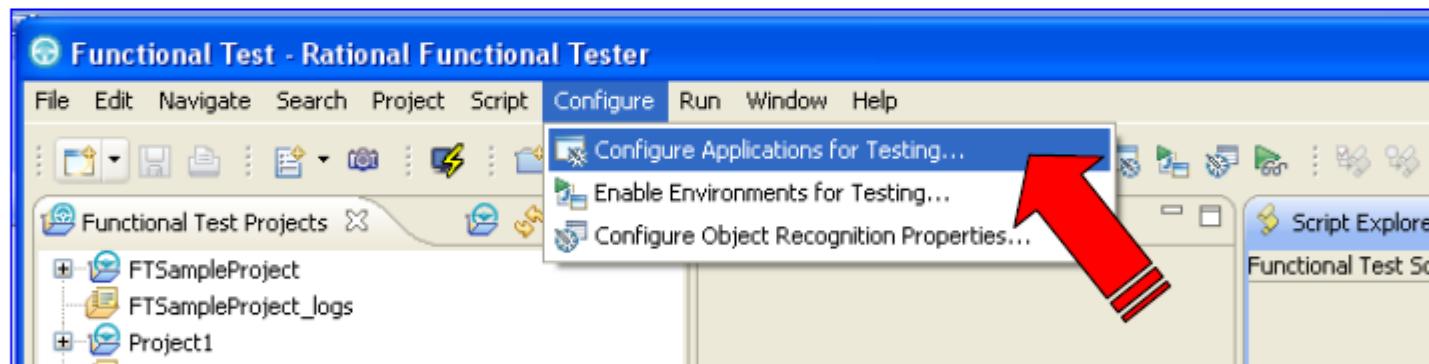
OPCIONAL

Habilitar plataformas de Eclipse

(página 61 del manual)



3. Configuración de aplicaciones Java a probar



Java Application
(Aplicaciones de escritorio)

(página 63 del manual)

HTML Application
(Aplicaciones Web)

(página 65 del manual)



TEMA 04: Script de Pruebas Funcionales

**Curso: Pruebas de software
Equipo de Profesores del Curso**

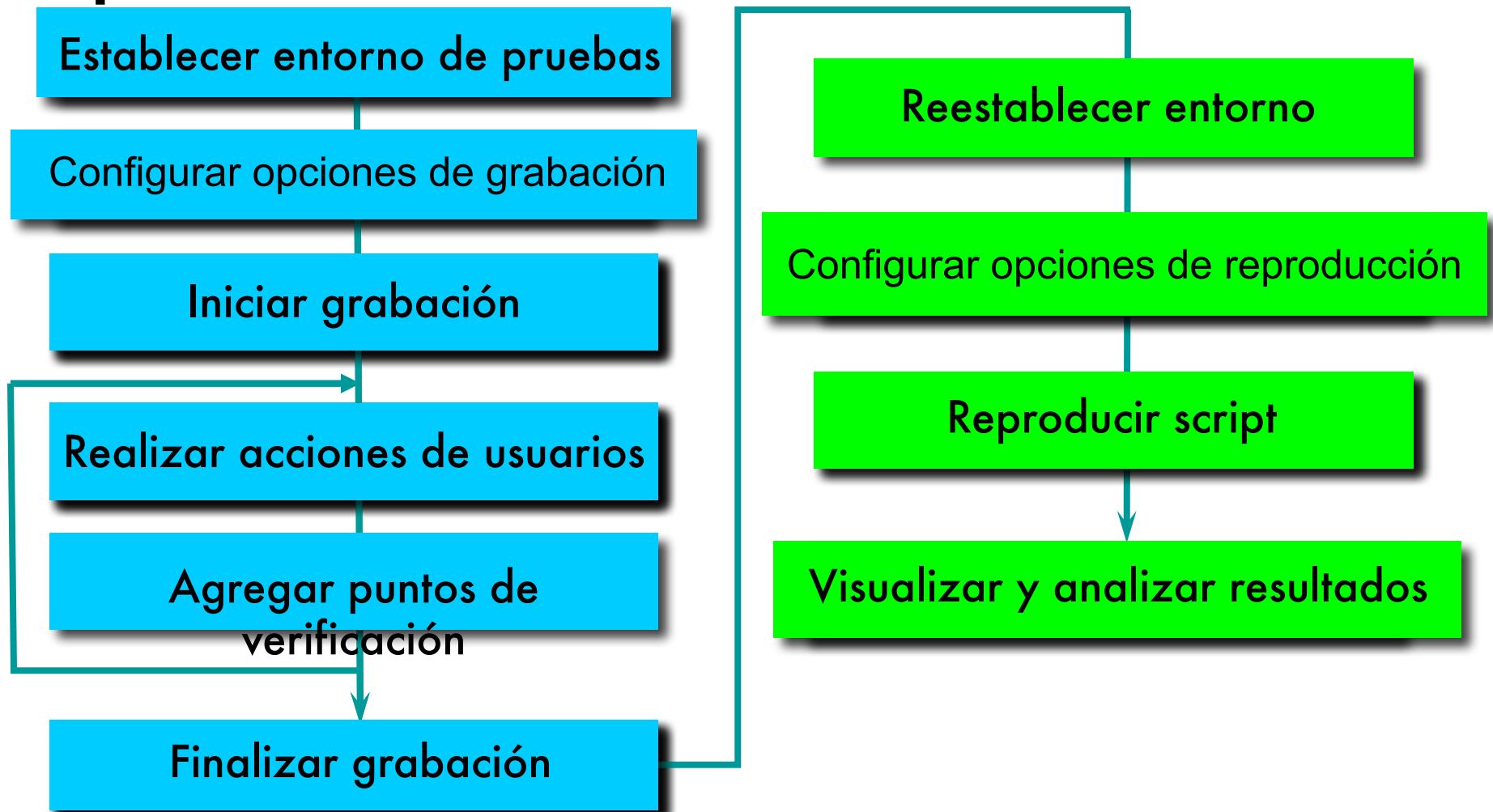


Temario

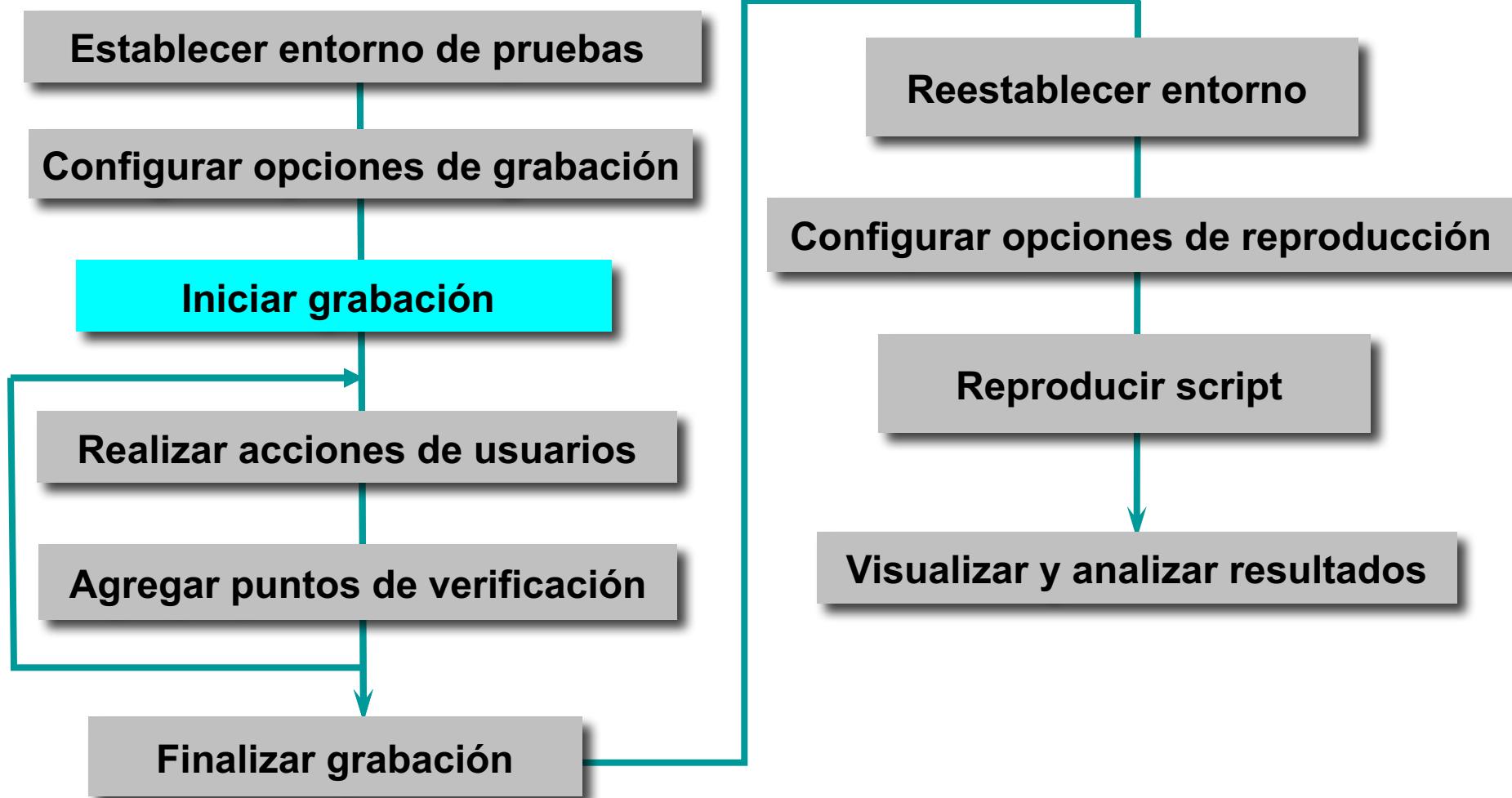
1. Proceso de grabación y reproducción de un script
2. Puntos de verificación

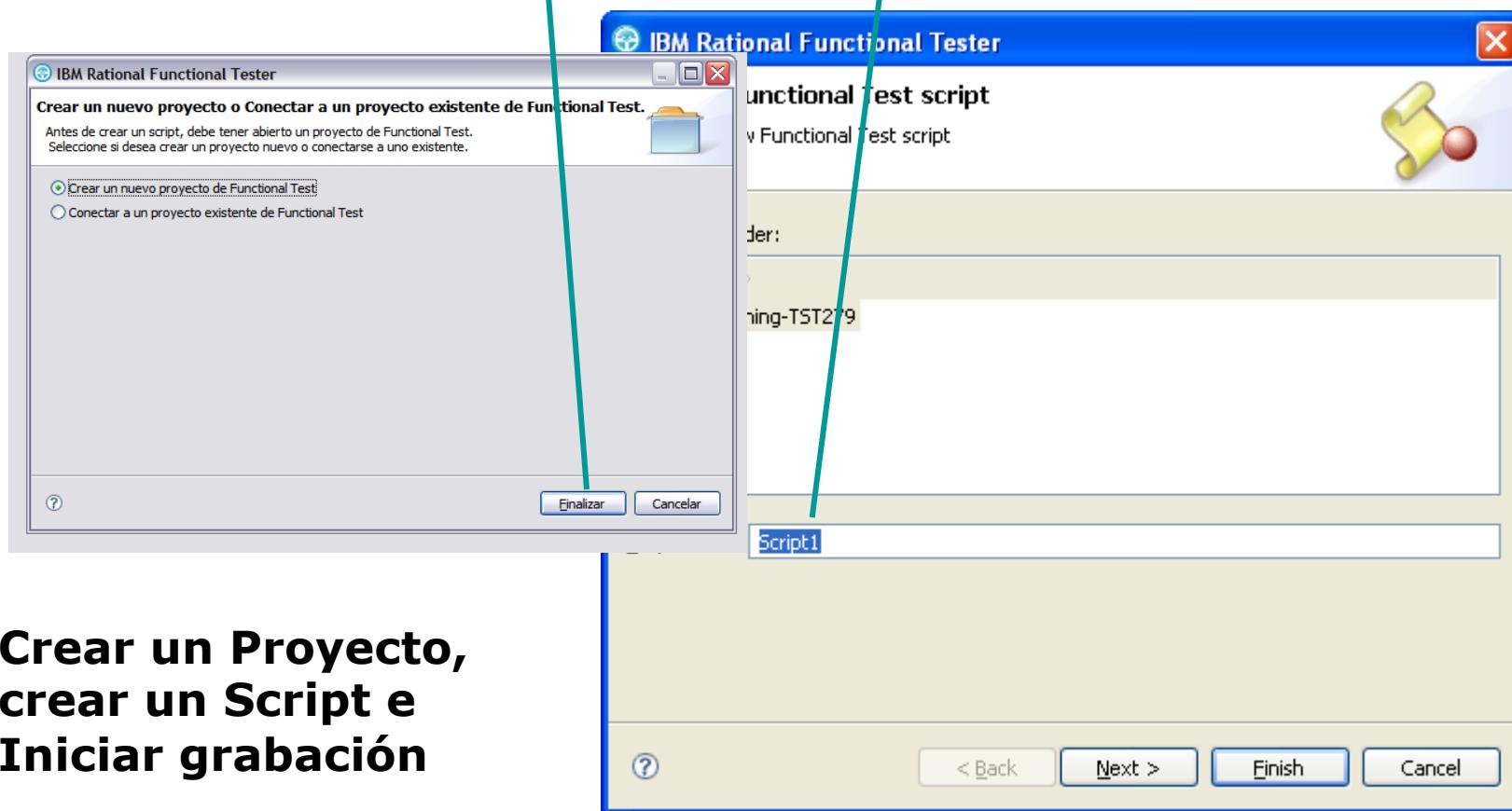


1. Proceso de grabación y reproducción de un script



Proceso de grabación y reproducción de un script





**Crear un Proyecto,
crear un Script e
Iniciar grabación**

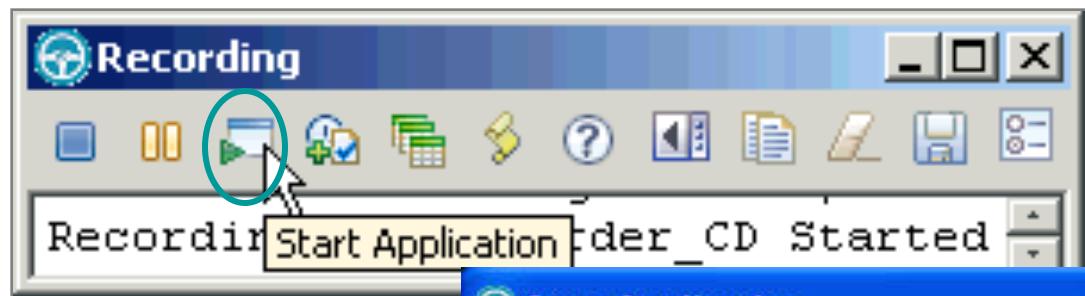


Barra de herramientas del monitor de grabación



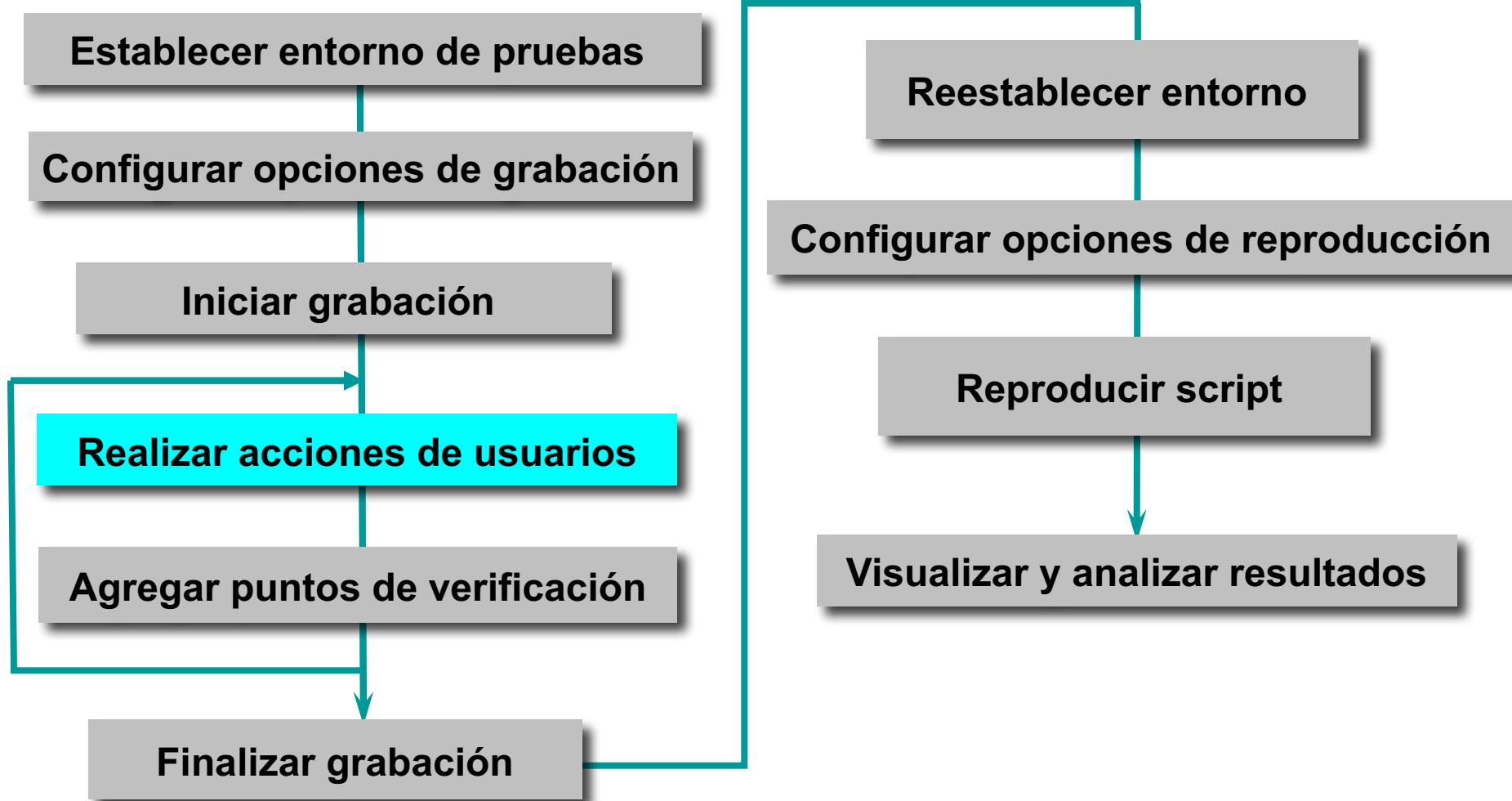
- 1
- 2
- 3
- 4
- 5
- 6
- 7

1. Detener Grabación
2. Pausa/Reanudar Grabación
3. Iniciar Aplicación
4. Agregar Puntos de Verificación
5. Agregar Comandos Controlados por Datos
6. Ayudas de Script
7. Mostrar ayudas



Iniciar Aplicación

Proceso de grabación y reproducción de un script





Realizar acciones de usuario

The screenshot illustrates a user interaction process across three windows:

- Step 1:** The left window shows a file tree under "Composers". The "Bach" folder is selected, highlighted with a yellow circle containing the number 1. It contains sub-folders "Brandenburg Concertos" and "Violin Concertos".
- Step 2:** A "Place Order" button is highlighted with a yellow circle containing the number 2.
- Step 3:** A "Member Logon" dialog box is open. The "Existing Customer" radio button is selected, highlighted with a yellow circle containing the number 3. The "Full Name" field contains "Trent Culpito" and the "Password" field contains "*****". Both fields are enclosed in a red box.
- Step 4:** The "OK" button in the "Member Logon" dialog is highlighted with a yellow circle containing the number 4.
- Step 5:** The right window shows the "Place an Order" screen. The "Item" field contains "Bach" and the "Quantity" field contains "1". The total price is \$15.99. The card information section is highlighted with a red box and contains "Card Number: 1234 1234 1234 1234", "Card Type: Visa", and "Expiration Date: 12/12". The name, address, city, state, zip, and phone fields are also visible.



Agregar comandos controlados por datos

The screenshot illustrates the process of adding data-driven commands to a software application. On the left, the 'Place an Order' screen shows various input fields for an order, including Item (Bach), Quantity (1), and Shipping & Handling (\$1.00). On the right, the 'Insert Data Driven Actions' dialog box is open, showing a table of data-driven commands. Step 1 highlights the 'Recording' toolbar icon. Step 2 highlights the dialog title. Step 3 highlights the 'Data Driven Commands' table, which lists items like 'Item' (Test Object) with command 'setText' and variable 'Composer' (Initial Value 'Bach'). Step 4 highlights the 'OK' button.

Role	Test Object	Command	Variable	Initial Value
Item	setText	Composer	Bach	
	1499	setText	Descripcion	Violin Concertos
Quantity	setText	Cantidad	1	
CardNumberIncl...	setText	NumTarjeta	1234 1234 1234...	
creditCombo	setText	TipoTarjeta	Visa	
ExpirationDate	setText	FechaExpiracion	12/12	
Name	setText	Nombre	Trent Culpito	
Street	setText	Direccionl	75 Wall St 22nd Fl	

Permitirá ingresar
datos de prueba
en un Pool de
Datos



Visualizar Pool de Datos

Functional Test - Proyecto1/Script1.rftss - Functional Functional Tester

File Edit Navigate Search Project Script Configure Window Help

1

2

Un Pool de Datos contendrá los datos de los casos de prueba

3

4

5

6 Clic sobre una celda y luego Enter para ingresar más datos

Script Java

Properties Tasks Console Problems Test Datapool

Private Test Datapool

	Composicion::ja...	Descripcion::ja...	Cantidad::java...	NumTarjeta::ja...	TipoTarjeta::ja...	FechaExpiracion::ja...	Nombre::java.l...	Direccion::java...
0	Bach	Violin Concertos	1	1234 1234 123...	Visa	12/12	Trent Culpito	75 Wall St 22nd Fl

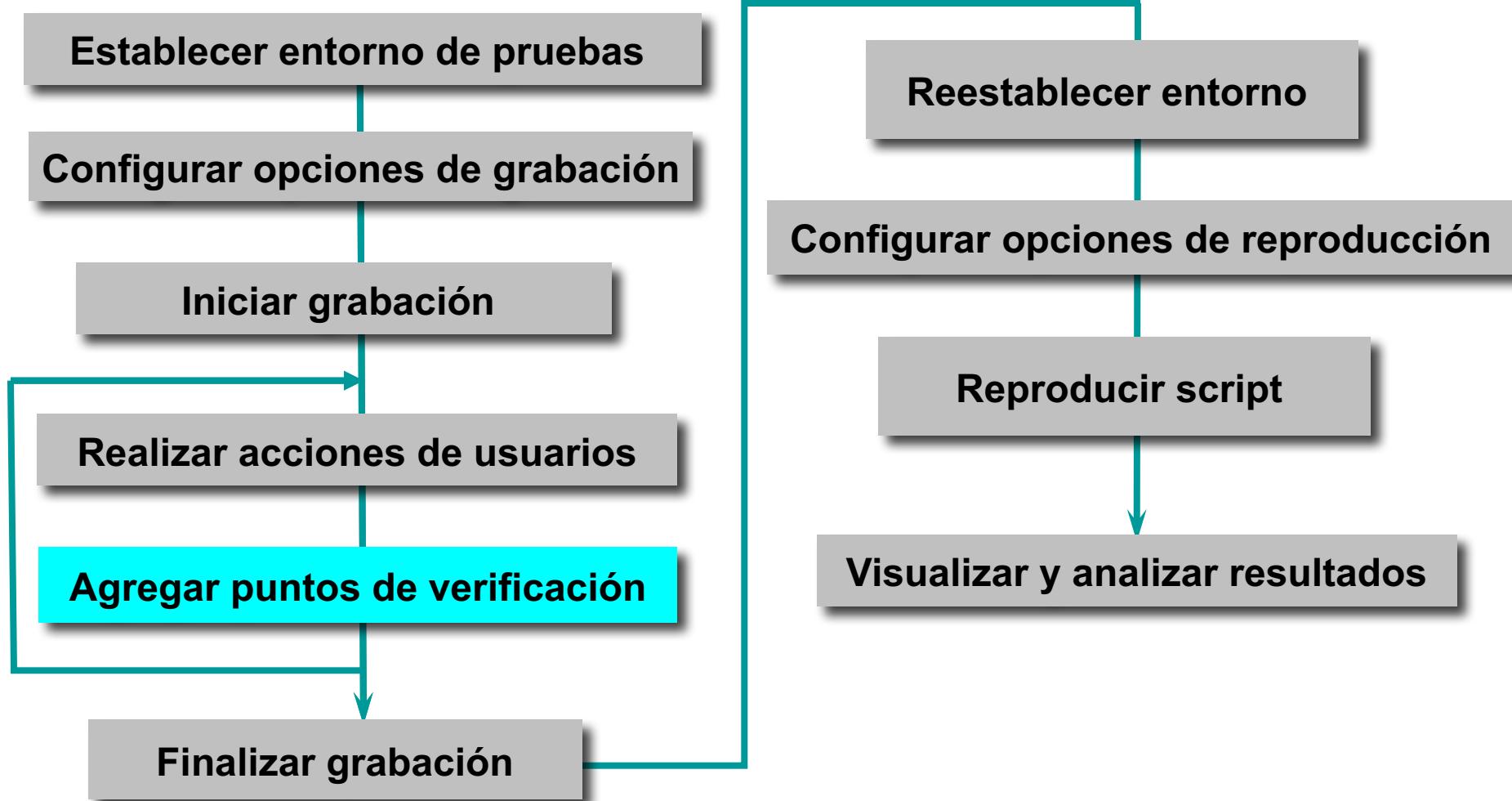
type filter text

- Data Management
- Debug
- Functional Test
 - Application View
 - Functional Test Projects
 - Functional Test Show Checkouts
 - Functional Test Show History
 - Keyword View
 - Script Explorer
 - Test Datapool
- Help
- Java
 - Java
 - Java Browsing
 - Jazz Source Control
- PDE
 - PDE
 - PDE Runtime

Use F2 to display the description for a selected view.

OK Cancel

Proceso de grabación y reproducción de un script

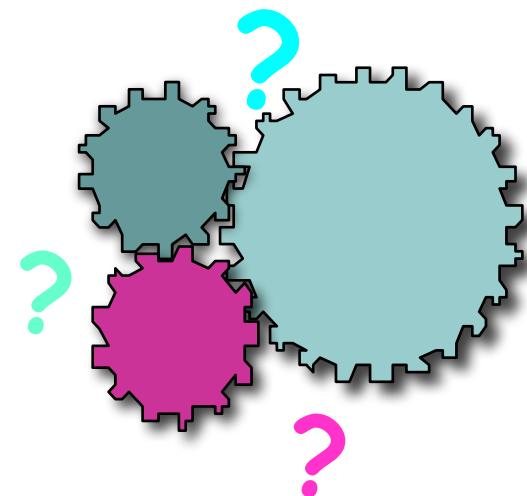


Puntos de Verificación (PV): Resultados de la aplicación

- ¿Cómo saber si la aplicación se ejecuta como lo esperamos?

Agregando PV, para ello grabar objetos que contengan:

- Sumas, totales, balances
- Datos cargados
- Mensajes de error
- Resultados de consultas

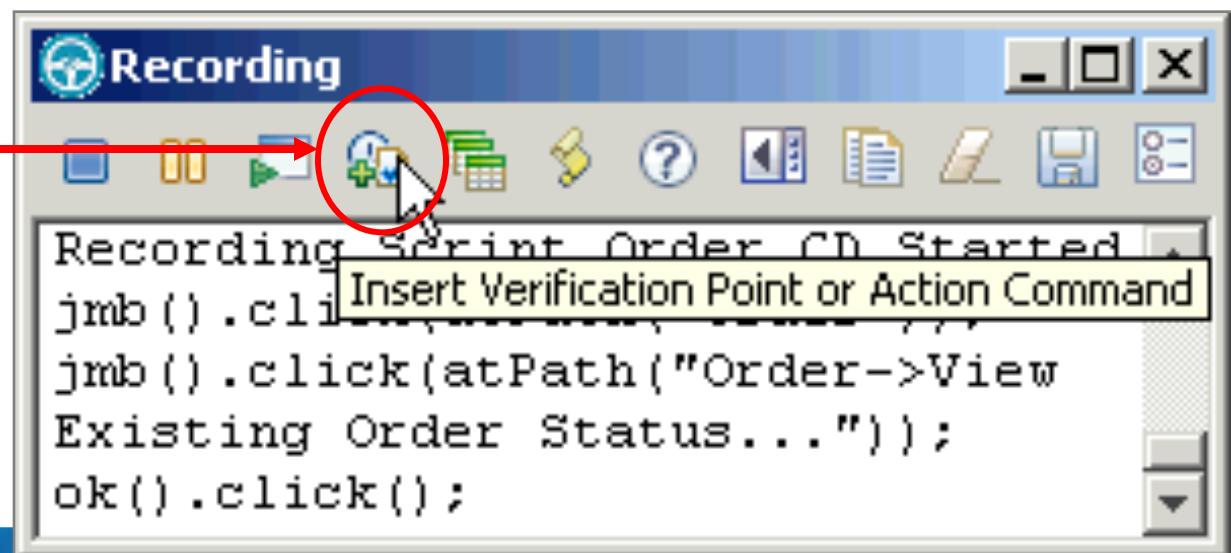




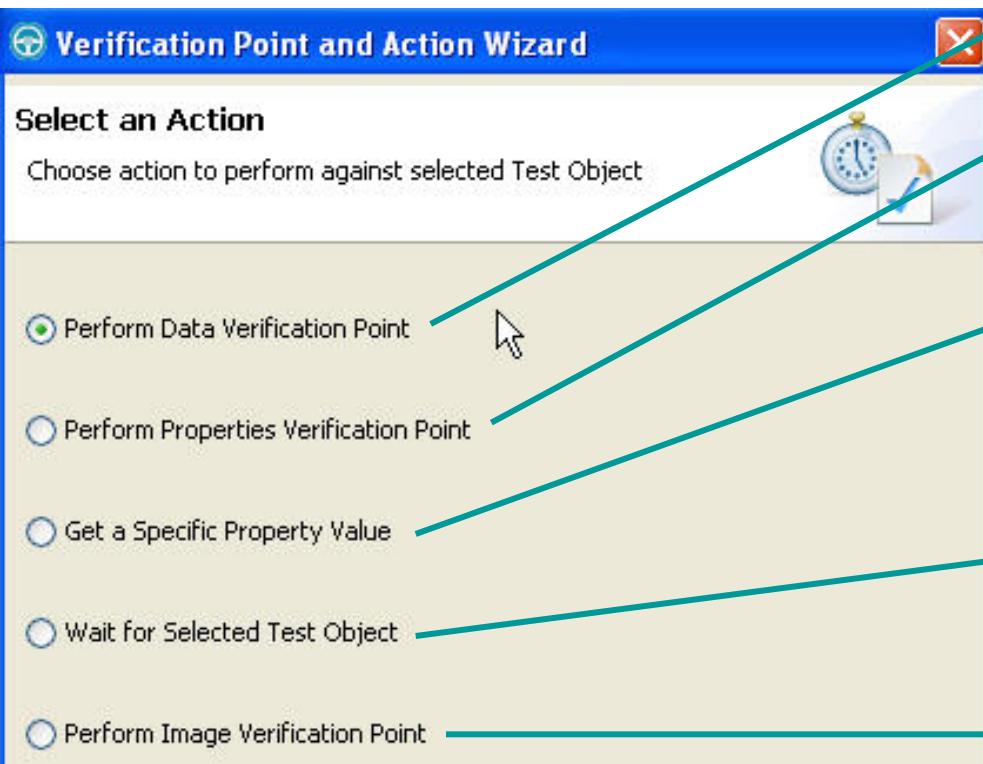
Puntos de Verificación: Dos pasos

1. Seleccionar un **objeto a probar** en la aplicación
2. Seleccionar una **acción a realizar** sobre el objeto

Para agregar
puntos de
verificación



Seleccionar una acción a realizar sobre el objeto

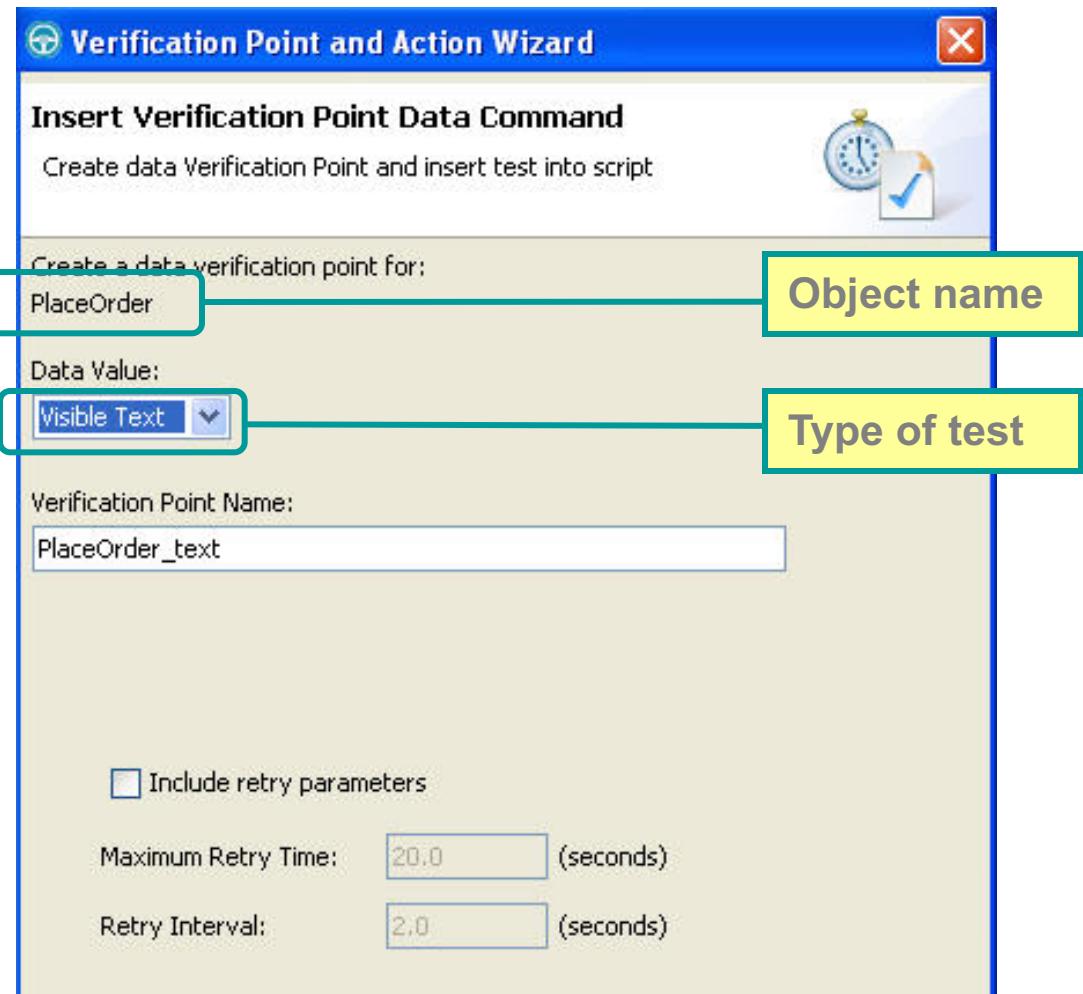


- ◆ Para verificar datos
- ◆ Para verificar propiedades del objeto
- ◆ Obtener un valor de una propiedad específica del objeto a probar
- ◆ Esperar a seleccionar el objeto a probar
- ◆ Visualizar cambios sobre una imagen

Seleccionar una Acción: Crear Puntos de Verificación para verificar datos

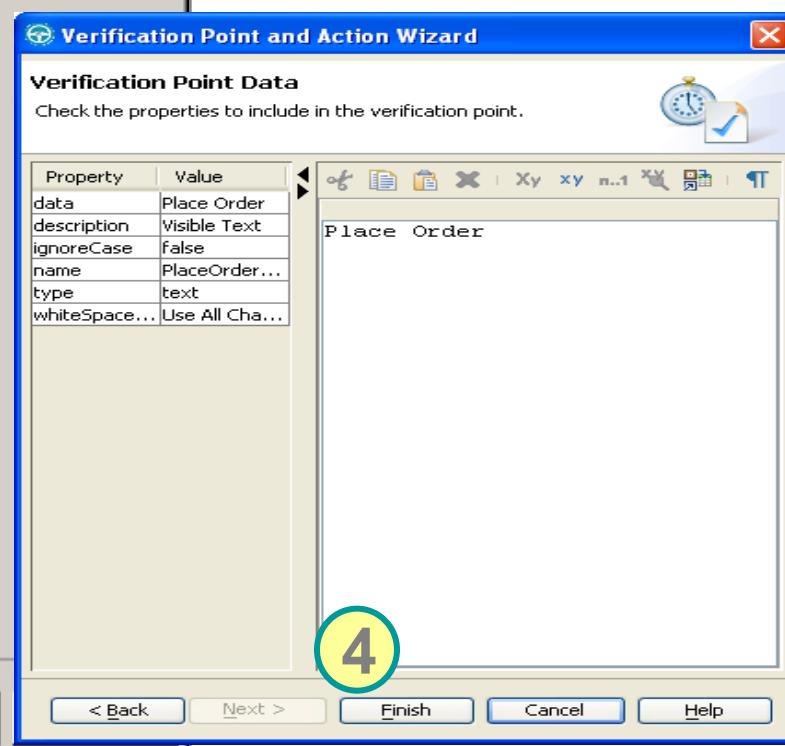
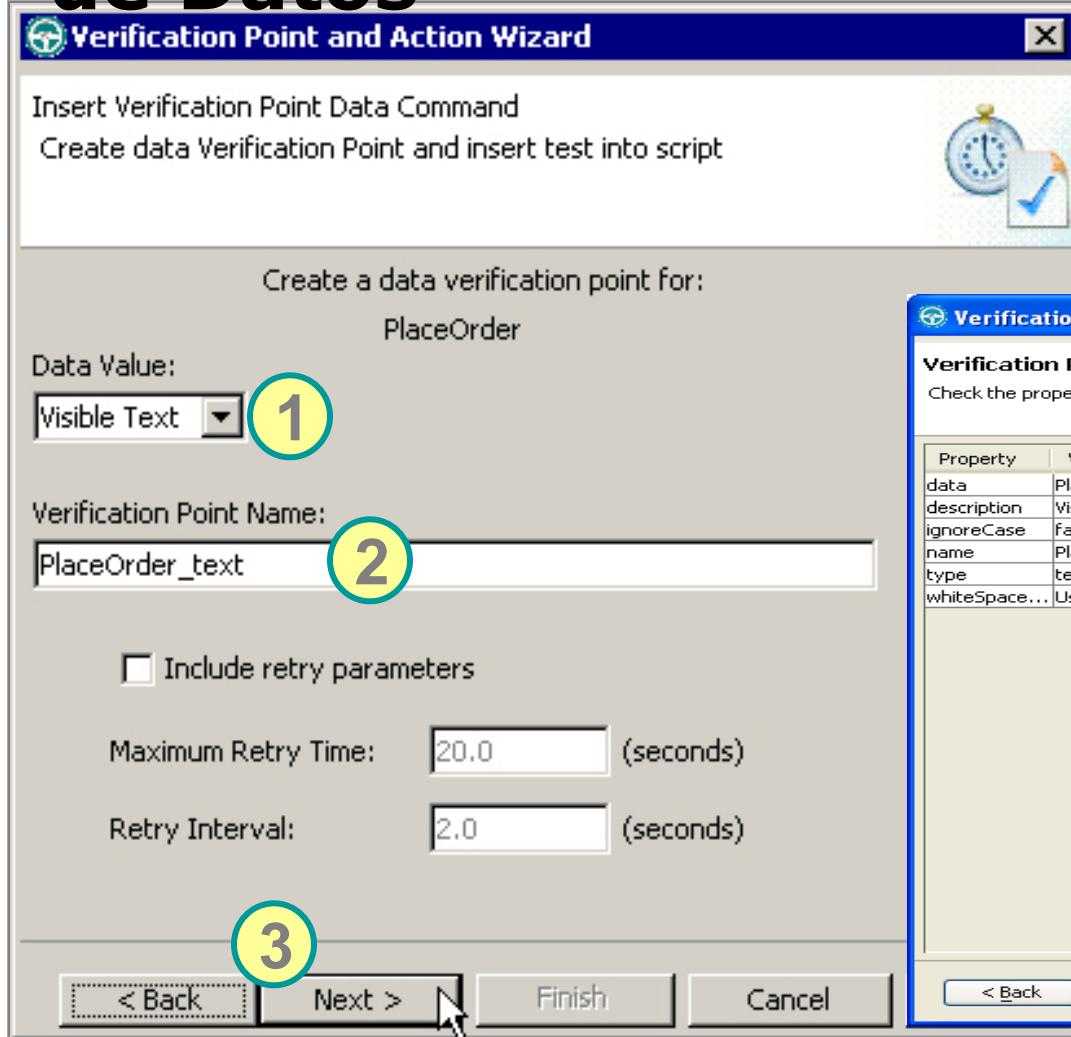
Tipos de datos que se pueden probar:

- Lista
- Menu
- Tablas
- Textos
- Estados



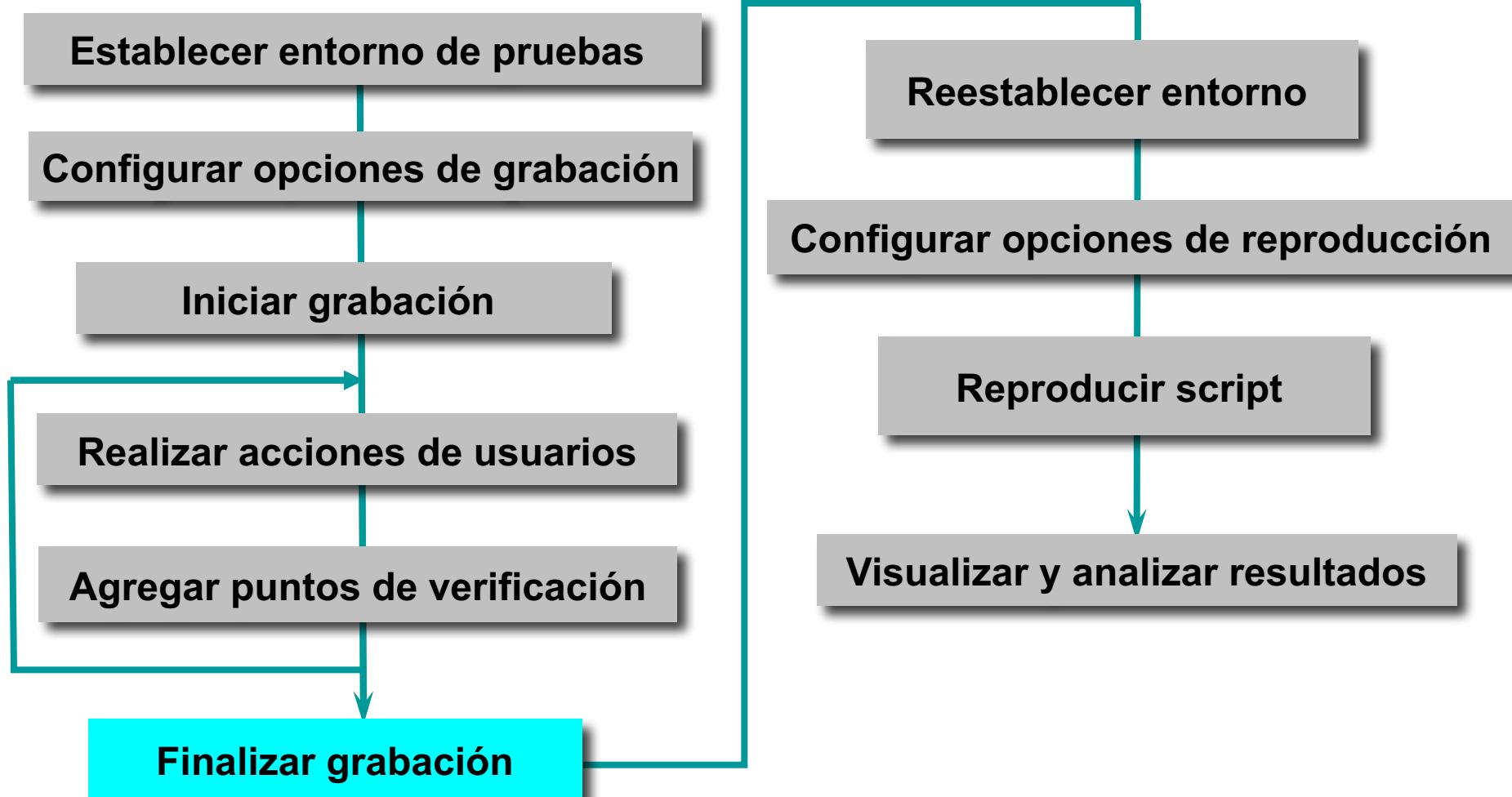


Ejemplo: Crear Puntos de Verificación de Datos





Proceso de grabación y reproducción de un script





Finalizar grabación

The screenshot shows the Rational Functional Tester interface. On the left, a 'Recording' window displays a Java script with several lines of code. One line, 'E Stop Recording der Status...'), is highlighted with a red box. Below the script is a tree view of recorded actions. On the right, the main Rational Functional Tester window shows the recorded script in a code editor. A yellow callout box points to the right window with the text 'Resultado: Un script de Java'. The Rational Functional Tester menu bar includes 'Configure', 'Run', 'Window', and 'Help'.

```
jFrame1.click(atPath("Order->View  
E Stop Recording der Status..."));  
ok().click();  
close().click();  
classicsJava(ANY, MAY_EXIT).close()  
);
```

MSG_OrderNewHaydnViolin_01
OrderTotal
PREF_OrderNewHaydn594_01
SCRIPTSUPPORT_OrderNewBachViolin
SharedMap
Simple_OrderNewSchubert55_01
Simple_OrderNewSchubertString_01
Simple_TCViewOrder_01
SimpleMap (Default)
TIMER_OrderNewSchubertString_02

Rational Functional Tester

Configure Run Window Help

Functional Test

```
// Data Driven Code inserted on Dec 27, 2008  
  
// Frame: Place an Order  
itemText().setText(dpString("Composer"));  
_1899Text().setText(dpString("Item"));  
quantityText().setText(dpString("Quantity"));  
cardNumberIncludeTheSpacesText().setText(dpSt:  
creditCombo().setText(dpString("CardType"));  
expirationDateText().setText(dpString("ExpDat:  
nameText().setText(dpString("Name"));  
t"));
```

Resultado: Un script de Java

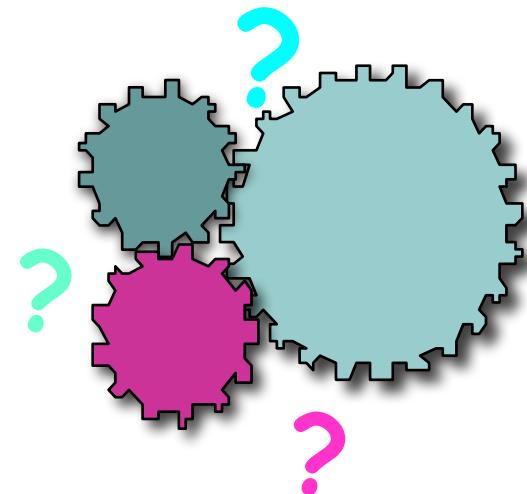
Tasks Console Problems Bookmarks

2. Puntos de Verificación (PV): Resultados de la aplicación

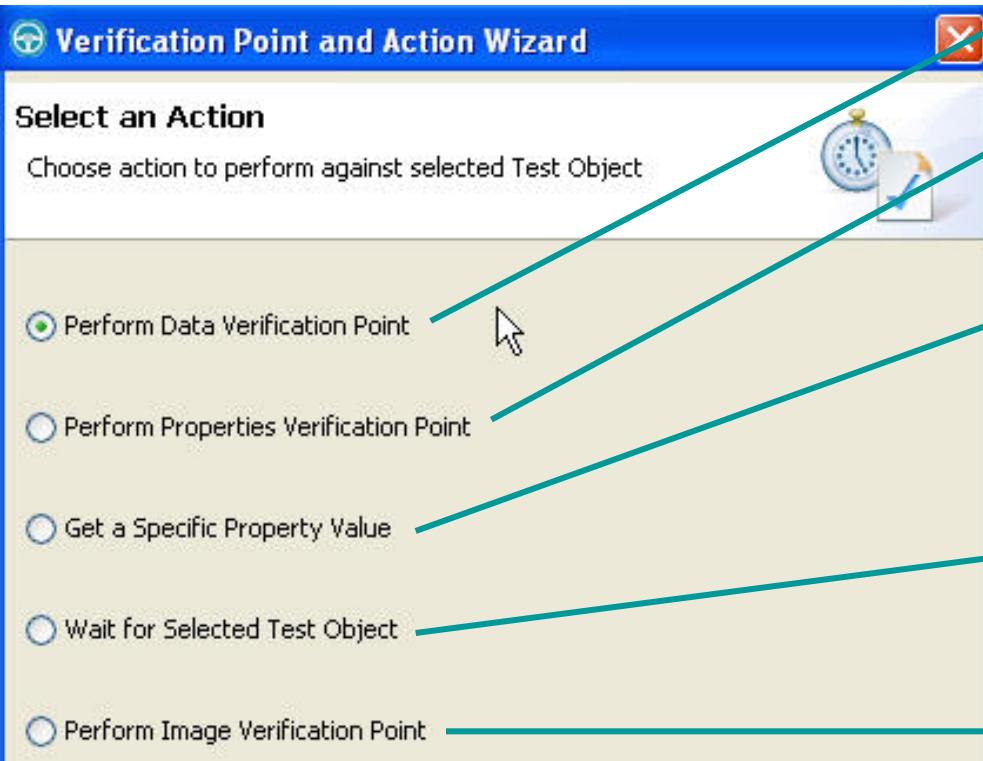
- ¿Cómo saber si la aplicación se ejecuta como lo esperamos?

Agregando PV, para ello grabar objetos que contengan:

- Sumas, totales, balances
- Datos cargados
- Mensajes de error
- Resultados de consultas
- Imágenes



Para grabar un PV: Seleccionar una acción a realizar sobre el objeto

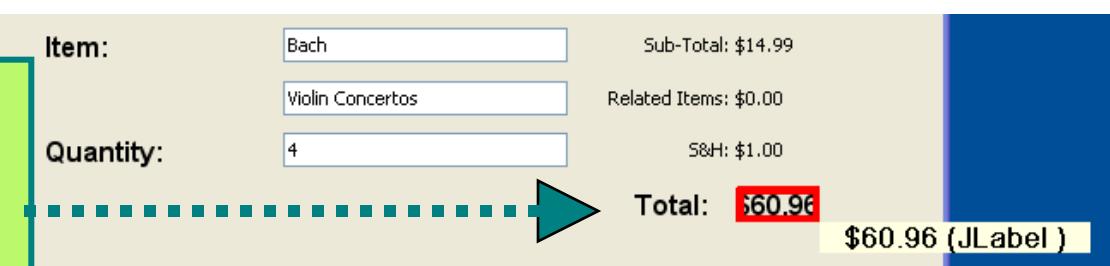


- 1 Para verificar datos
- 2 Para verificar propiedades del objeto
- 3 Obtener un valor de una propiedad específica del objeto a probar
- 4 Esperar a seleccionar el objeto a probar
- 5 Visualizar cambios sobre una imagen

1 PV para verificar datos

- Para probar los datos que se visualizan en la aplicación. Ejemplos:
 - Etiquetas que muestran los datos de un usuario identificado en el sistema.
 - Cadenas sobre campos de textos desactivados que muestran los datos cargados de un cliente.
 - Etiquetas que muestran valores como suma total, subtotales, etc.

Aquí se muestra una **etiqueta** con el valor de un monto total





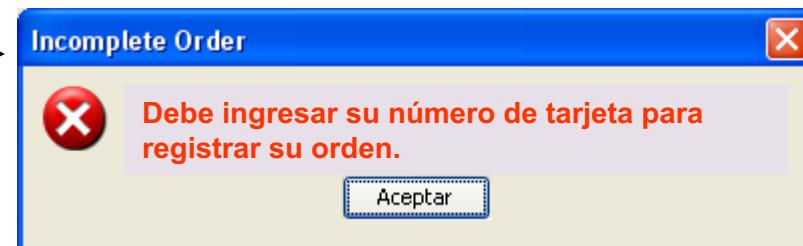
2

PV para verificar propiedades

Para probar las propiedades de un objeto en la aplicación. Ejemplos:

- El texto y el tamaño de la tabla (HTML_Table) que contiene un mensaje de error.
- El texto y color del mensaje de error sobre una etiqueta.

Aquí se muestra un **mensaje de error en color rojo**

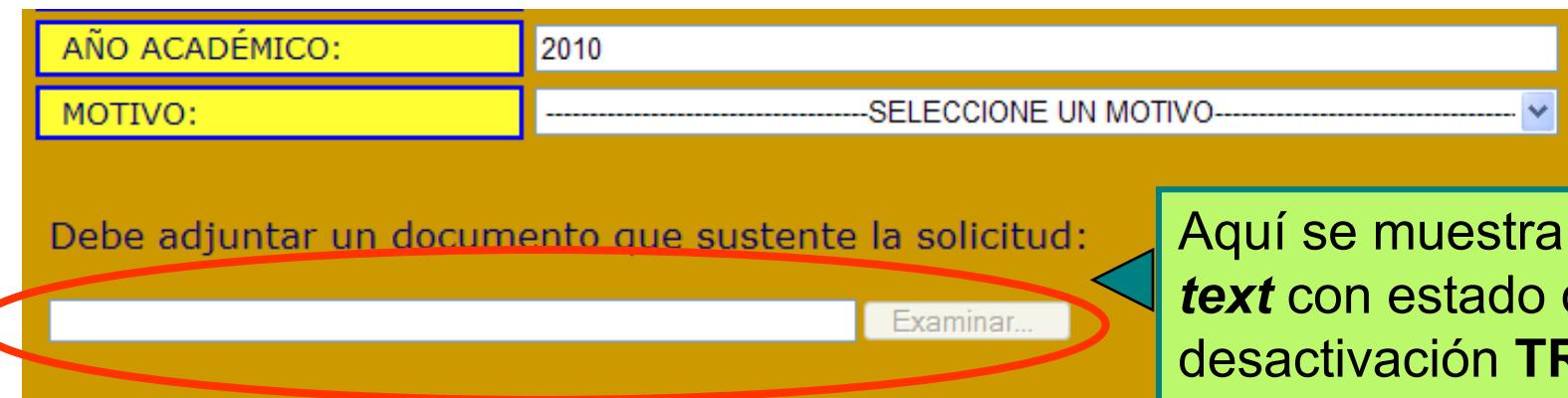


3

PV para obtener un valor de una propiedad específica

⊕ Para probar un valor de una propiedad específica de un objeto en la aplicación. Ejemplos:

- El valor del estado de desactivación de un **file text**



AÑO ACADÉMICO: 2010

MOTIVO: SELECCIONE UN MOTIVO

Debe adjuntar un documento que sustente la solicitud:

Examinar...

Aquí se muestra un **file text** con estado de desactivación **TRUE**. Este valor es **FALSE** si se seleccione un motivo.



3

PV para obtener un valor de una propiedad específica (Cont.)

⊕ Ejemplos:

- El valor del título de una página web que muestra un formulario con datos cargados

Aquí se muestra una página con un formulario de datos cargados.

The screenshot shows a web browser window with the title 'Result Form'. Inside the window, there is a list of form fields with their corresponding values. A red circle highlights this list. The fields and values are as follows:

firstname:	John
lastname:	Doe
street:	Tranquillo Prosperi
city:	Campinas
state / province:	Sao Paulo
zip:	13084778
phone:	1234556789
email:	doe@testing.com

URL de este ejemplo completo:

<http://www.ibm.com/developerworks/ssa/rational/library/10/automateintegrationtestswithrationalfunctionaltester/index.html>



PV para esperar a seleccionar el objeto a probar

- ⊕ Esta opción se utiliza en caso sea necesario especificar un tiempo de espera para visualizar el resultado que se cargará sobre un objeto. Esto es para evitar problemas en la visualización de resultados. Ejemplos:
 - Datos que se cargan en una lista desplegable (combo o HTML input type select) o en una tabla (HTML table) después de un filtro anterior.



PV para verificar cambios sobre una imagen

- ⊕ Para probar imágenes en la aplicación. Al grabar el punto de verificación, se crea un archivo de imagen de línea base. Cada vez que reproduce el script, la imagen se comparara para ver si se han producido cambios, ya sea de forma intencionada o no intencionada. **La verificación de la imagen se hace estrictamente píxel a píxel.** Esto resulta útil para identificar errores posibles.